案例 3-双缓冲动画算法

文档编写：霍波魏

校稿/修订：孔令德

时间 2019~2020

联系方式：QQ997796978

说明：本套案例由孔令德开发，原版本为 Visual C++6.0，配套于孔令德的著作《计算机图形学-基于 MFC 三维图形开发》一书。孔令德计算机工程研究所的学生霍波魏在学习计算机图形学期间，对本套案例进行了升级并编写了学习文档。现在程序的编写和程序的解释都是基于 Windows 10 操作系统，使用 Microsoft visual studio 2017 平台的 MFC（英文版）开发。

一、知识点

1. 双缓冲绘图

一次性将图形绘制到内存缓冲区，然后使用源拷贝方式将图形拷贝到设备缓冲区。

2. 定时器

为程序设置定时器定时器，每隔一定时间间隔调用双缓冲绘图，绘图时改变.bmp 图片中每个位图的位置。

二、案例描述

在资源中添加了一个有 10 个位图的长图，通过双缓冲将位图做成一个连续不断无闪烁的动画。

三、实现步骤

1. 添加位图资源，长图的 ID 编号为 IDB_DRAGON。
2. 添加 DoubleBuffer()双缓冲函数
3. 在双缓冲函数中通过调用 LoadBitmap 函数以将加载的视图附加到 CBitmap 对象
4. 状态栏的修改以及对 bPlay 进行设置

四、主要算法

1. CTestView 类

```
void CTestView::DoubleBuffer(CDC* pDC)//双缓冲
{
```

```
CRect rect;
GetClientRect(&rect);
CDC memDC;//内存设备上下文，用于绘制前景
memDC.CreateCompatibleDC(pDC);
CBitmap NewBitmap, *pOldBitmap;
NewBitmap.LoadBitmap(IDB_DRAGON);
pOldBitmap = memDC.SelectObject(&NewBitmap);
BITMAP bmp;
NewBitmap.GetBitmap(&bmp);
int nBmpHeight = bmp.bmHeight / m_TotalBmps;
pDC->BitBlt((rect.Width() - bmp.bmWidth) / 2, (rect.Height() - nBmpHeight) / 2,
    rect.Width(), nBmpHeight, &memDC, 0, m_Num * nBmpHeight, SRCCOPY);
                        //将背景拷贝到显示器屏幕上
memDC.SelectObject(pOldBitmap);
NewBitmap.DeleteObject();
memDC.DeleteDC();
}
```

五、实现效果

龙的双缓冲动画效果如图 3-1 所示。

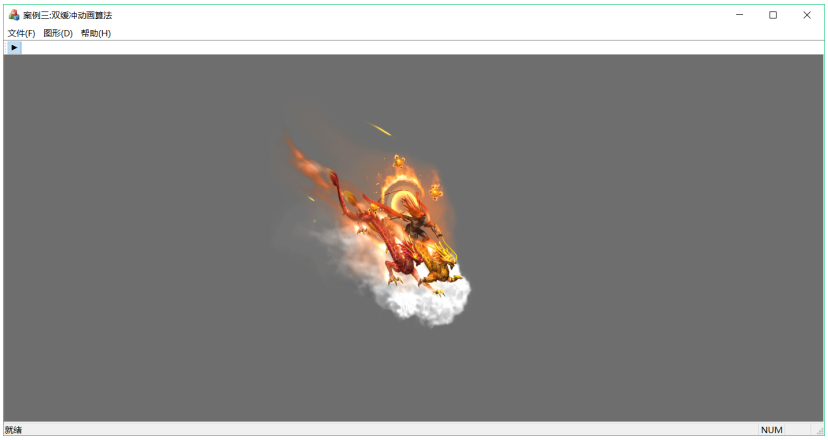

图 3-1 双缓冲动画效果图

六、遇到的问题及解决方案

1. 添加启动图标按钮

通过设置菜单命令和 Commond 命令来控制。

2. 改变位图位置

在类向导种添加 WM_TIMER 消息映射函数中改变.bmp 图片中每个位图的位置。

3. 动画闪烁问题

m_Num = ++m_Num % (m_TotalBmps)解决了闪烁问题，原来直接对位图取

余是位图从 0-12，之后从 1-12，就会造成屏幕闪烁，先++再取余则是第二次以至于后面都是从 0-12，这样就从位图循环上消除了动画闪烁。

七、案例心得

在本案例中将原程序中两次将背景拷贝到显示器屏幕上，在本程序中将其优化，直接在双缓冲中绘制。该案例中还解决了双缓冲动画闪烁的问题，解决该问题最关键的一点是添加了一个消息响应函数，将屏幕背景设置放到消息响应函数中，是设置过程从窗体转移到了后台，从而提升了运行效率，消除了动画闪烁。